**CRTC INTERCONNECTION STEERING COMMITTEE**

**CONTRIBUTION FORM:**

**Working Group:            CSCN                                  Date of Submission: 2024-05-07**

**Contribution #: 242A**

**TIF #:      118                                                                   File ID: CNCO242A**

**Task Title: Update CSCN-Administered Guidelines for Thousands-Block Pooling**

**Related to Task(s) ID: 118**

**Contributor:**

**Name: Ed Antecol**

**Company: COMsolve Inc.**

**Address:**

**Tel:**

**Fax:**

**E-mail: edward.antecol@comsolveinc.com**

**Distribution to: CSCN**

**Subject: LSMS and SCP functionality in a TBP environment**

This contribution discusses the necessary review, by service providers, of the following network elements/functions prior to the implementation of Thousands-Block pooling:

1. Local Service Management System (LSMS);
2. Service Control Point (SCP) LNP Application; and
3. GTT.

Further to the action item with respect to changing LNP data in a Thousands-Block pooling environment, for clarification, the structure of the LNP “dip” will not change. However, the upstream data that is used to determine the LNP “dip” response will change as discussed below.

**Background**

GTT Global Title Translation

IAM Initial Address Message

ISDN Integrated Services Digital Network

ISUP ISDN User Part

NPAC Number Portability Administration Center

NPDB Number Portability Database

SCP Service Control Point

SS7 Signaling System 7

SSP Service Switching Point

STP Signal Transfer Point

The term SCP is used in this contribution in a generic sense to imply an element in the network that runs the LNP application (and possibly other applications such as GTT). This contribution is not intended to imply any specific platform requirements (e.g., Bellcore's SCP requirements, GR-1280-CORE).

Information about ported subscribers and active Thousands-Blocks is made available from the NPAC to LSMSs which in turn make this information available to SCPs (typically via a local NPDB) supporting the LNP Application

The LSMS function may be part of the SCP LNP Application or may reside elsewhere. A single LSMS may support multiple SCPs. A provider may also use more than one LSMS to serve a given SCP LNP Function in order to provide greater reliability. Specific configurations between LSMS systems and LNP SCPs are up to each individual service provider. The following diagram illustrates two different configurations by which a service provider’s switching systems may utilize for NPAC sourced data.



General assumption for supporting LNP Queries

When a customer ports their service to a different service provider, while retaining their TN, the two service providers (old and new) supply information to the NPAC.

The SCP LNP Application will look up the dialed TN typically from within a local database that is populated by a feed from an LSMS, and if the number has been ported, the SCP will return a response including a Location Routing Number (LRN) of the recipient switch (i.e., switch hosting the ported number). When the switch receives the LRN (from the LNP “dip”), the LRN will be used to route the call to its correct destination.

In a legacy switching architecture, the switch will use the received LRN to populate the ISUP IAM's Called Party Number parameter. The actual dialed digits (plus implied Area Code) are placed in the Generic Address Parameter (GAP) of the ISUP IAM.

Queries for non-ported TNs will cause the SCP to return the actual dialed TN and not an LRN. In this case, the SSP (switch) will route the call based on analysis of the dialed TN. The SCP LNP Application need not know whether the SSP generating the LNP query is the originating switch, an intermediate/tandem switch, or a terminating/donor switch.

**Issue 1: Testing LSMSs to Ensure Proper Configuration to Receive and Process Thousands-Block Records**

Pre-Thousands-Block Pooling, the set of data which must be populated in the LSMS consists of individual ported TN record (each instance is called a subscription version).

The NPAC SMS issues an M-CREATE for the subscriptionVersion to each of the Local SMSs, that is accepting downloads for the NPA-NXX of the subscriptionVersion. For the XML interface, SVCD – SVCreateDownload. See Appendix A and note that the LNPtype is equal to “LSPP" or "LISP".

In a Thousands-Block Pooling environment, the LSMSs must support receipt of Thousands-Block record information from the NPAC. For each Thousands-Block that is activated,[[1]](#footnote-1) there are two parts to the data from the NPAC to the LSMSs. The first part is the “header” portion, called NPA-NXX-X (or “dash-X”), and the second part is the “detail” portion, called the Number Pool Block.

NPAC step #1 to send an NPA-NXX-X broadcast message to LSMSs.

The NPAC SMS sends an M-CREATE request for the serviceProvNPA-NXX-X object to all Local SMSs who support the object according to the “NPAC Customer LSMS NPA-NXX-X Indicator” in their service provider profile on the NPAC SMS and are receiving data for the NPA-NXX. For the XML interface, DXCD – NpaNxxDxCreateDownload. The following attributes are sent in the M-CREATE:

serviceProvNPA-NXX-X-ID

serviceProvNPA-NXX-X-Value

serviceProvNPA-NXX-X-CreationTimeStamp

serviceProvNPA-NXX-X-EffectiveTimeStamp

serviceProvNPA-NXX-X-ModifiedTimeStamp

serviceProvNPA-NXX-X-DownloadReason

NPAC step #2 to send a number pool block broadcast message to LSMSs.

NPAC SMS issues the M-CREATE for the numberPoolBlock to the Local SMS. For the XML interface, PBCD – NpbCreateDownload.

See Appendix A for message details.

**Issue 2: SCP LNP Application Compatibility with Thousands-Block Pooling**

The SCP LNP Applications (which access locally stored LSMS data) must support Thousands-Block records received from the LSMS.

**Issue 3: GTT Information**

Canadian operators may need to confirm whether existing SCP LNP GTT functions and the GTT distribution process (<https://www.cnac.ca/other_codes/ss7/ss7_network_codes.htm> ) will continue to work in a Thousands-Block Pooling environment, and if not, what changes are required.

**Appendix A**

Subscription Version Data

For a CMIP-based LSMS, this is the data that is sent.

SubscriptionDownloadData ::= SET OF SEQUENCE {

 subscription-version-id [0] SubscriptionVersionId,

 subscription-version-tn [1] PhoneNumber OPTIONAL,

 subscription-data SubscriptionData

}

SubscriptionData ::= SEQUENCE {

 subscription-lrn [1] LRN OPTIONAL,

 subscription-new-current-sp [2] ServiceProvId OPTIONAL,

 subscription-activation-timestamp [3] GeneralizedTime OPTIONAL,

 subscription-class-dpc [4] EXPLICIT DPC,

 subscription-class-ssn [5] EXPLICIT SSN,

 subscription-lidb-dpc [6] EXPLICIT DPC,

 subscription-lidb-ssn [7] EXPLICIT SSN,

 subscription-isvm-dpc [8] EXPLICIT DPC,

 subscription-isvm-ssn [9] EXPLICIT SSN,

 subscription-cnam-dpc [10] EXPLICIT DPC,

 subscription-cnam-ssn [11] EXPLICIT SSN,

 subscription-end-user-location-value [12]

 EndUserLocationValue OPTIONAL,

 subscription-end-user-location-type [13] EndUserLocationType OPTIONAL,

 subscription-billing-id [14] BillingId OPTIONAL,

 subscription-lnp-type [15] LNPType,

 subscription-download-reason [16] DownloadReason,

 subscription-wsmsc-dpc [17] EXPLICIT DPC OPTIONAL,

 subscription-wsmsc-ssn [18] EXPLICIT SSN OPTIONAL,

 subscription-sv-type [19] EXPLICIT SVType OPTIONAL,

 subscription-optional-data [20] EXPLICIT OptionalData OPTIONAL

}

For an XML-based LSMS, this is the data that is sent.

 <xs:complexType name="SubscriptionVersionCreateData">

 <xs:sequence>

 <xs:element name="range\_notif\_tn\_id\_info" type="RangeNotifyTnIdInfo"/>

 <xs:element name="svb\_lrn" type="Lrn"/>

 <xs:element name="svb\_new\_sp" type="ServiceProvId"/>

 <xs:element name="svb\_activation\_timestamp" type="xs:dateTime"/>

 <xs:element name="svb\_class\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_class\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_lidb\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_lidb\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_isvm\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_isvm\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_cnam\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_cnam\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_end\_user\_location\_value" type="EndUserLocationValue" minOccurs="0"/>

 <xs:element name="svb\_end\_user\_location\_type" type="EndUserLocationType" minOccurs="0"/>

 <xs:element name="svb\_billing\_id" type="BillingId" minOccurs="0"/>

 <xs:element name="sv\_lnp\_type" type="LNPType"/>

 <xs:element name="download\_reason" type="DownloadReason"/>

 <xs:element name="svb\_wsmsc\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_wsmsc\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_sv\_type" type="SVType" minOccurs="0"/>

 <xs:element name="svb\_optional\_data" type="OptionalData" minOccurs="0"/>

 </xs:sequence>

 </xs:complexType>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NPA-NXX-X Data

For a CMIP-based LSMS, this is the data that is sent.

NPA-NXX-X-DownloadData ::= SET OF SEQUENCE {

 service-prov-npa-nxx-x-id [0] NPA-NXX-X-ID,

 service-prov-npa-nxx-x [1] NPA-NXX-X OPTIONAL,

 service-prov-npa-nxx-x-effective-timestamp [2] GeneralizedTime OPTIONAL,

 service-prov-npa-nxx-x-creation-timestamp [3] GeneralizedTime OPTIONAL,

 service-prov-npa-nxx-x-modified-timestamp [4] GeneralizedTime OPTIONAL,

 service-prov-npa-nxx-x-download-reason [5] DownloadReason

}

For an XML-based LSMS, this is the data that is sent.

<xs:complexType name="NpaNxxXData">

 <xs:sequence>

 <xs:element name="sp\_id" type="ServiceProvId"/>

 <xs:element name="npa\_nxx\_x\_id" type="NpaNxxXId"/>

 <xs:element name="npa\_nxx\_x\_value" type="NpaNxxX"/>

 <xs:element name="npa\_nxx\_x\_effective\_timestamp" type="xs:dateTime"/>

 <xs:element name="npa\_nxx\_x\_creation\_timestamp" type="xs:dateTime"/>

 <xs:element name="npa\_nxx\_x\_modified\_timestamp" type="xs:dateTime"/>

 <xs:element name="download\_reason" type="DownloadReason"/>

 </xs:sequence>

 </xs:complexType>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Number Pool Block Data

For a CMIP-based LSMS, this is the data that is sent.

NumberPoolBlock-CreateAction ::= SEQUENCE {

 block-npa-nxx-x NPA-NXX-X,

 block-holder-sp ServiceProvId,

 block-lrn LRN,

 block-class-dpc DPC,

 block-class-ssn SSN,

 block-lidb-dpc DPC,

 block-lidb-ssn SSN,

 block-isvm-dpc DPC,

 block-isvm-ssn SSN,

 block-cnam-dpc DPC,

 block-cnam-ssn SSN,

 block-wsmsc-dpc [0] DPC OPTIONAL,

 block-wsmsc-ssn [1] SSN OPTIONAL,

 block-sv-type [2] SVType OPTIONAL,

 block-optional-data [3] OptionalData OPTIONAL

}

For an XML-based LSMS, this is the data that is sent.

<xs:complexType name="NumberPoolBlockData">

 <xs:sequence>

 <xs:element name="block\_id" type="BlockId"/>

 <xs:element name="block\_dash\_x" type="NpaNxxX"/>

 <xs:element name="sp\_id" type="ServiceProvId"/>

 <xs:element name="svb\_activation\_timestamp" type="xs:dateTime"/>

 <xs:element name="svb\_lrn" type="Lrn"/>

 <xs:element name="svb\_class\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_class\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_lidb\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_lidb\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_isvm\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_isvm\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_cnam\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_cnam\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="svb\_wsmsc\_dpc" type="Dpc" minOccurs="0"/>

 <xs:element name="svb\_wsmsc\_ssn" type="Ssn" minOccurs="0"/>

 <xs:element name="download\_reason" type="DownloadReason"/>

 <xs:element name="svb\_sv\_type" type="SVType" minOccurs="0"/>

 <xs:element name="svb\_optional\_data" type="OptionalData" minOccurs="0"/>

 </xs:sequence>

 </xs:complexType>

1. Thousands-Blocks associated with the CO Code Holder can rely on NPA-NXX routing and therefore do not require a Thousands-Block record activation. An exception to this is when the CO Code Holder requires the Thousands-Block to exist on a different switch than the CO Code. [↑](#footnote-ref-1)